Treeview

A hierarchical tree view component that provides an intuitive way to display and navigate nested data structures. Built with accessibility in mind, it supports expandable/collapsible nodes, customizable icons, multiple size variants, and comprehensive keyboard navigation for building file browsers, navigation menus, and organizational charts.

How to use

Note : The TreeView component is standalone and includes all necessary dependencies for common modules and Lucide icons.

Basic Usage

Simple tree view with expandable nodes and basic selection.

```
import { AavaTreeviewComponent, TreeNode } from "@aava/play-core";
```

Features

Hierarchical Structure

- Nested Nodes : Support for unlimited nesting levels
- Expandable/Collapsible : Interactive nodes that can be expanded or collapsed
- Dynamic Indentation : Automatic indentation based on node level
- Recursive Rendering : Self-referential component for nested structures

Visual Customization

- Multiple Sizes : Five size variants (xs, sm, md, lg, xl)
- Icon Positioning : Left or right-aligned expand/collapse controls
- Custom Icons : Support for Lucide icons and folder states
- Responsive Design : Adapts to different screen sizes

User Interaction

- Node Selection : Click to select individual nodes
- Keyboard Navigation : Full keyboard support for accessibility
- Expand/Collapse : Click toggle controls or use arrow keys

- Hover States : Visual feedback for interactive elements

Accessibility

- ARIA Support : Proper ARIA attributes for screen readers
- Keyboard Navigation : Arrow keys, Enter, and Space for interaction
- Focus Management : Clear focus indicators and logical tab order
- Semantic Structure : Proper HTML semantics for tree navigation

API Reference

Inputs

Property	Type	Default	Description
nodes	TreeNode[]	[]	Array of tree nodes to display
size	'xs' | 'sm' | 'md' | 'lg' | 'xl'	'md'	Size variant for the tree nodes
iconPosition	'left' | 'right'	'left'	Position of expand/collapse controls
level	number	0	Current nesting level (used internally)

Outputs

Event	Type	Description
nodeSelect	EventEmitter<TreeNode>	Emitted when a node is selected

Methods

Method	Parameters	Return	Description
toggleExpand()	node: TreeNode	void	Toggle the expanded state of a node

Method	Parameters	Return	Description
selectNode()	node: TreeNode	void	Select a node and emit selection event
calculateIndent()	level?: number	number	Calculate indentation for a given level
handleKeyDown()	event: KeyboardEvent, node: TreeNode	void	Handle keyboard navigation events

Interfaces

TreeNode

Focus Management

- Each tree node is focusable with tabindex="0"
- Toggle controls have tabindex="-1" to prevent tab navigation
- Focus indicators provide clear visual feedback
- Logical tab order follows the tree structure

Design Tokens & Theming

AAVA Play TreeView uses semantic design tokens for all surfaces, spacing, and typography. The component exposes scoped override tokens for fine-tuning appearance while maintaining design system consistency.

Available Design Tokens for TreeView

Node Tokens

Token	Purpose	Default Value
--tree-node-gap	Gap between node elements	Theme-based
--tree-node-height-xs	Extra small node height	Theme-based

Token	Purpose	Default Value
--tree-node-height-sm	Small node height	Theme-based
--tree-node-height-md	Medium node height	Theme-based
--tree-node-height-lg	Large node height	Theme-based
--tree-node-height-xl	Extra large node height	Theme-based
--tree-node-font-weight-xl	Font weight for extra large	Theme-based
--tree-node-line-height-xs	Line height for extra small	Theme-based
--tree-node-line-height-medium	Line height for medium	Theme-based
--tree-node-line-height-lg	Line height for large	Theme-based
--tree-node-line-height-xl	Line height for extra large	Theme-based

Toggle Control Tokens

Token	Purpose	Default Value
--tree-toggle-size-xs	Extra small toggle width	Theme-based
--tree-toggle-size-sm	Small toggle width	Theme-based
--tree-toggle-size-md	Medium toggle width	Theme-based
--tree-toggle-size-lg	Large toggle width	Theme-based
--tree-toggle-size-xl	Extra large toggle width	Theme-based

Icon Tokens

Token	Purpose	Default Value
--tree-icon-size-xs	Extra small icon size	Theme-based
--tree-icon-size-sm	Small icon size	Theme-based
--tree-icon-size-lg	Large icon size	Theme-based
--tree-icon-size-xl	Extra large icon size	Theme-based

Label Tokens

Token	Purpose	Default Value
--tree-label-font-family	Font family for labels	Theme-based
--tree-label-font-size-xs	Extra small font size	Theme-based
--tree-label-font-size-sm	Small font size	Theme-based
--tree-label-font-size-medium	Medium font size	Theme-based
--tree-label-font-size-lg	Large font size	Theme-based
--tree-label-font-size-xl	Extra large font size	Theme-based

Color Tokens

Token	Purpose	Default Value
--color-text-primary	Primary text color	Theme-based
--rgb-brand-disabled	Brand color for states	Theme-based

Token Override Example

Best Practices

Design Guidelines

- Consistent Hierarchy : Use consistent indentation and visual cues
- Clear Labels : Ensure node names are descriptive and concise
- Appropriate Icons : Use meaningful icons that represent node types
- Size Selection : Choose size variants that match your content density
- Icon Positioning : Consider user expectations for expand/collapse controls

Accessibility

- Keyboard Navigation : Ensure all interactions work with keyboard
- Screen Reader Support : Provide clear labels and descriptions
- Focus Indicators : Maintain visible focus states

- ARIA Attributes : Use proper ARIA roles and properties
- Color Contrast : Ensure sufficient contrast for text and icons

Performance

- Lazy Loading : Consider lazy loading for large tree structures
- Virtual Scrolling : Implement virtual scrolling for very large trees
- Change Detection : Use OnPush strategy for better performance
- Memory Management : Clean up event listeners and references

User Experience

- Visual Feedback : Provide clear hover and selection states
- Smooth Animations : Use transitions for expand/collapse actions
- Consistent Behavior : Maintain predictable interaction patterns
- Error Handling : Gracefully handle invalid data structures

Integration

- Data Structure : Ensure your data follows the TreeNode interface
- Event Handling : Implement proper selection and expansion logic
- State Management : Coordinate tree state with your application
- Styling : Use design tokens for consistent theming

Responsive Behavior

Mobile Adaptations

The tree view component automatically adapts to mobile screens:

- Touch Optimization : Appropriate touch targets for mobile interaction
- Mobile Layout : Optimized spacing and sizing for small screens
- Gesture Support : Touch-friendly expand/collapse interactions
- Responsive Icons : Icon sizes that work well on mobile

Breakpoint Behavior

- Desktop (>768px) : Full tree interface with all features
- Mobile (≤768px) : Compact layout with optimized spacing
- Node Display : Responsive node sizing and spacing
- Icon Scaling : Appropriate icon sizes for different screens

Content Considerations

- Node Names : Node labels adapt to different screen widths
- Indentation : Appropriate indentation levels for mobile
- Icon Visibility : Icons remain visible and accessible
- Touch Targets : Adequate touch target sizes for mobile

Use Cases

File System Navigation

- File Browsers : Navigate through directory structures
- Document Management : Organize and browse documents
- Media Libraries : Browse photo and video collections
- Code Repositories : Navigate project file structures

Organizational Charts

- Company Structure : Display organizational hierarchy
- Team Management : Show team relationships and roles
- Project Structure : Organize project components
- Category Management : Display product or content categories

Navigation Systems

- Website Navigation : Site structure and menu systems
- Application Menus : App navigation and settings
- Breadcrumb Navigation : Hierarchical navigation paths
- Sitemap Display : Website structure visualization

Data Visualization

- Hierarchical Data : Display nested data relationships
- Taxonomy Systems : Show classification hierarchies
- Decision Trees : Visualize decision-making processes
- Workflow Diagrams : Display process flows and steps